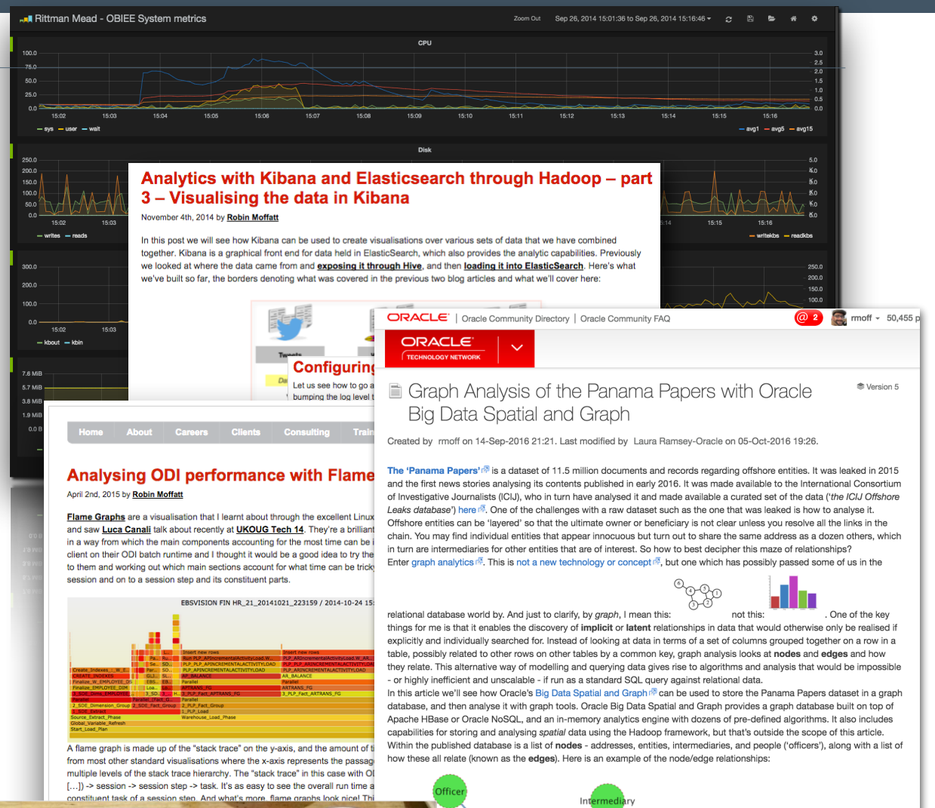
Source Control, Code Deployment, and Concurrent Development for OBIEE 12c

UKOUG Tech16

Robin Moffatt, Rittman Mead

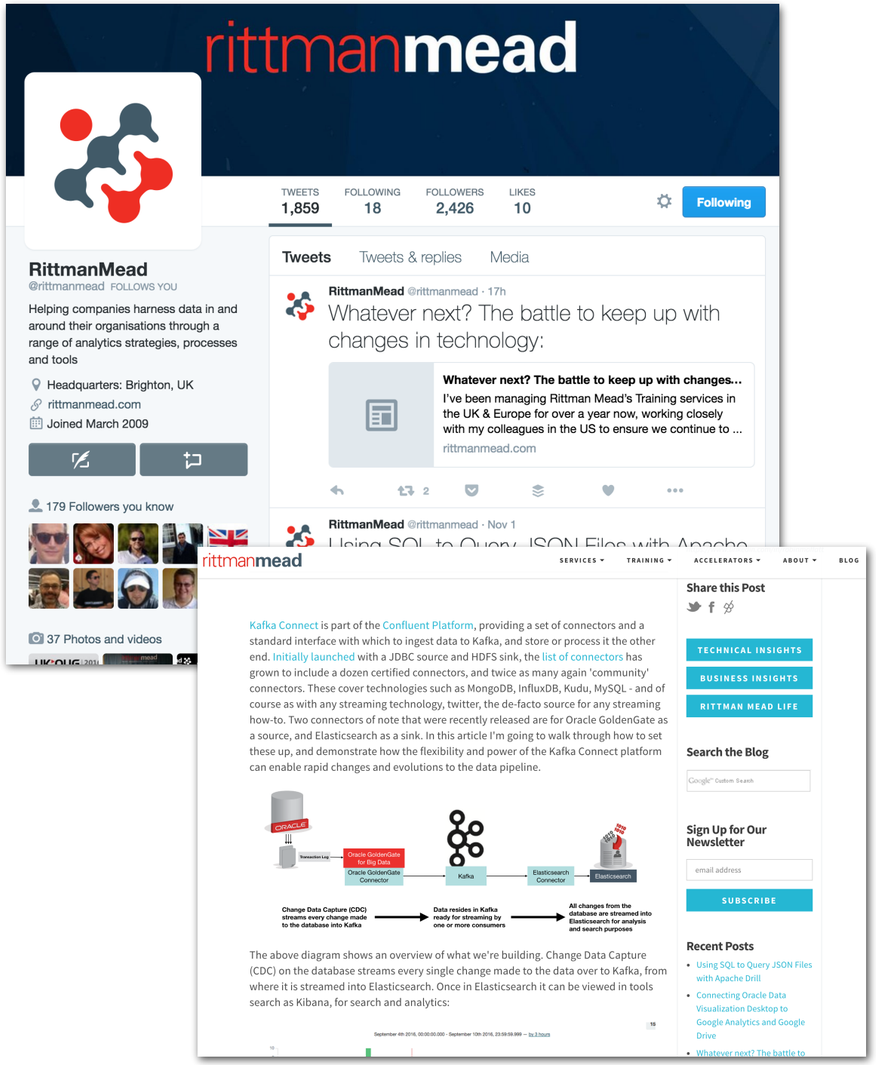
Robin Moffatt

- Head of R&D (Europe), Rittman Mead
  - Previously OBIEE/DW developer at large UK retailer
  - Previously SQL Server DBA, Business Objects, DB2, COBOL….
- Oracle ACE
- Frequent blogger : http://ritt.md/rmoff and http://rmoff.net
- Twitter: @rmoff
- IRC: rmoff / #obihackers / freenode



Rittman Mead

- Oracle Gold Partner with offices in the UK and USA
- 70+ staff delivering Oracle BI, DW, Big Data and Advanced Analytics projects
- 1 Oracle ACE Director + 2 Oracle ACEs
- Significant web presence with the Rittman Mead Blog (http://www.rittmanmead.com)
- Hadoop R&D lab for “dogfooding” solutions developed for customers



OBIEE Software Development Life Cycle (SDLC)

- A successful OBIEE implementation is more than just data modelling and analytics.

- Software Development LifeCycle (SDLC) is of great importance to all involved in OBIEE:
  - Developers
  - Release Managers
  - Testing
  - Project & Delivery Management

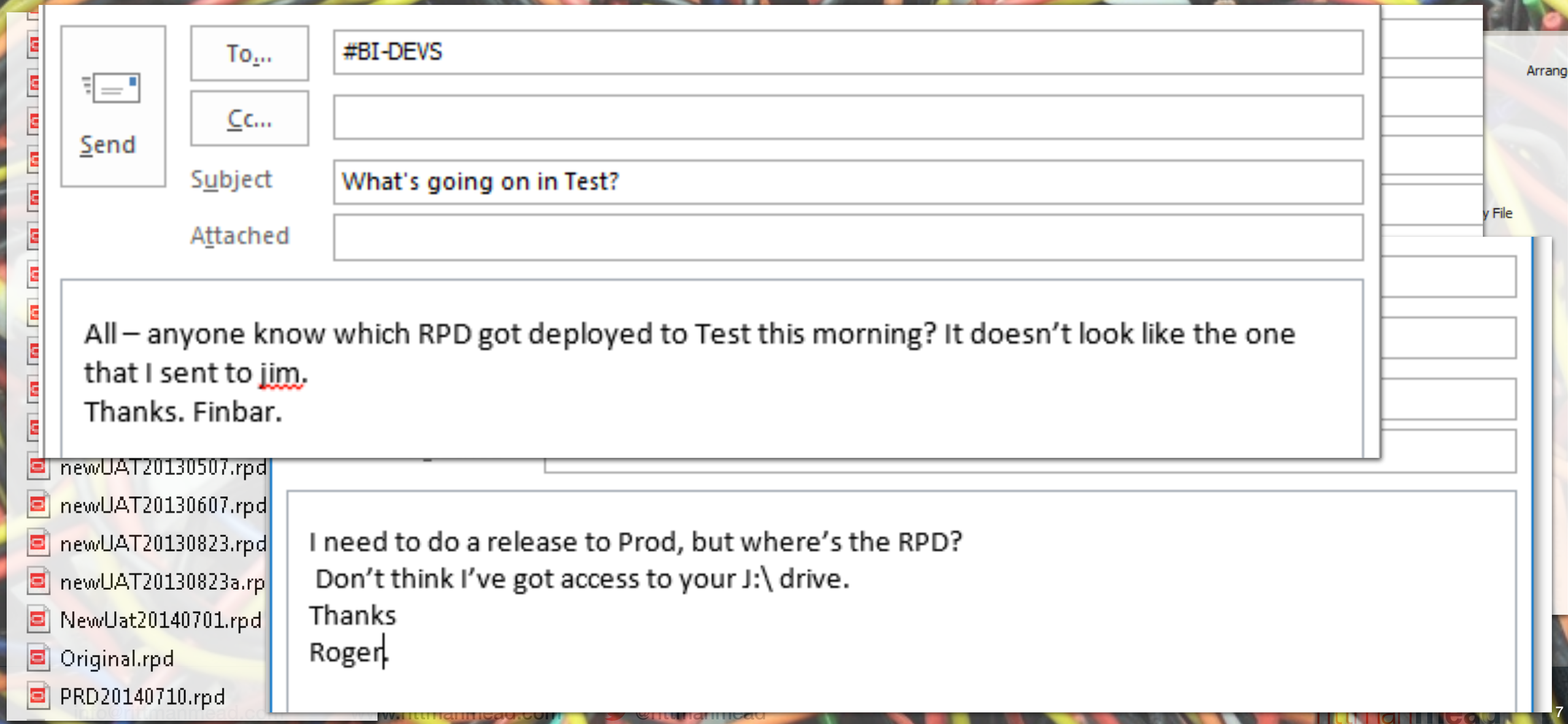
OBIEE Software Development Life Cycle (SDLC)

- Key areas of importance:
  - Source Control
  - Code Deployment
  - Concurrent Development

Source Control in OBIEE

(Source Control == Revision Control == Version Control)

Without Source Control We Have Chaos!



Why Do We Need Source Control?

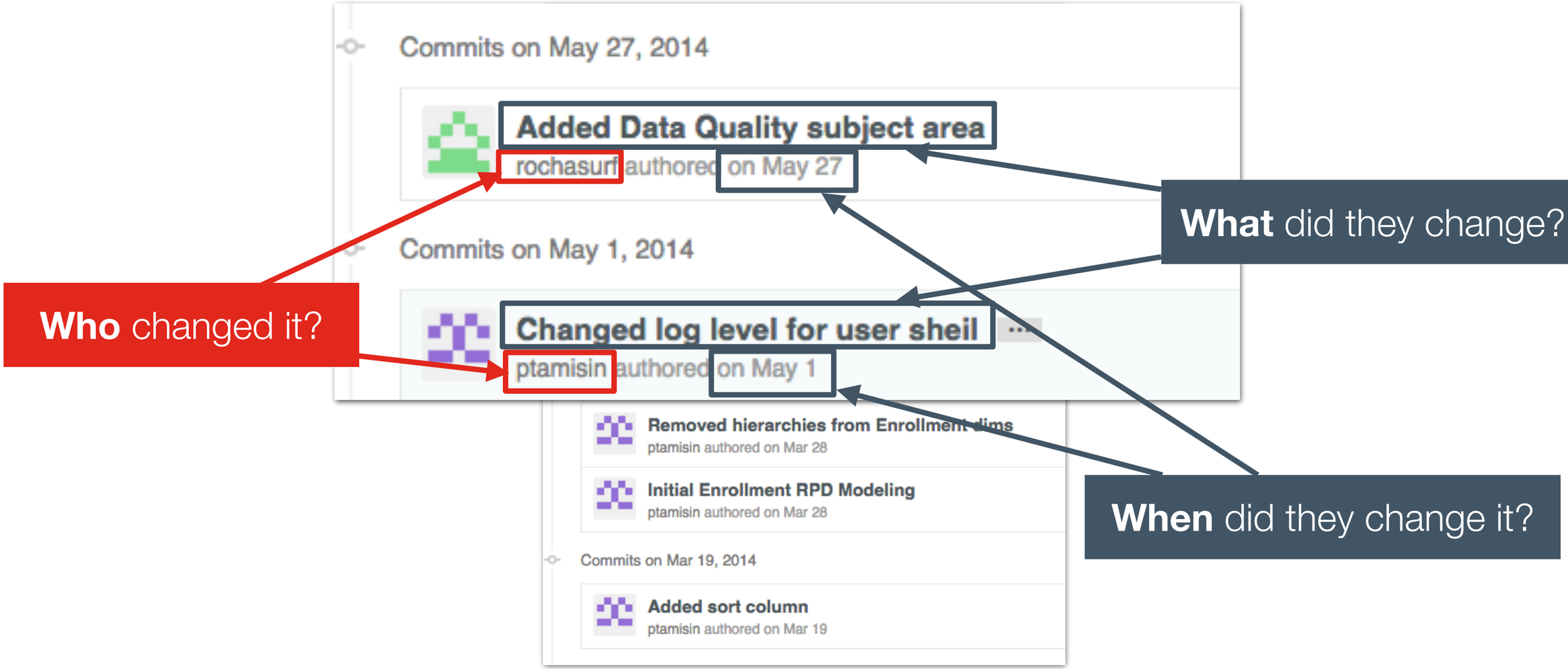
- **Every** mature OBIEE development should be using Source Control!

  - Reduce risk when deploying

  - Improve ease of maintenance and development

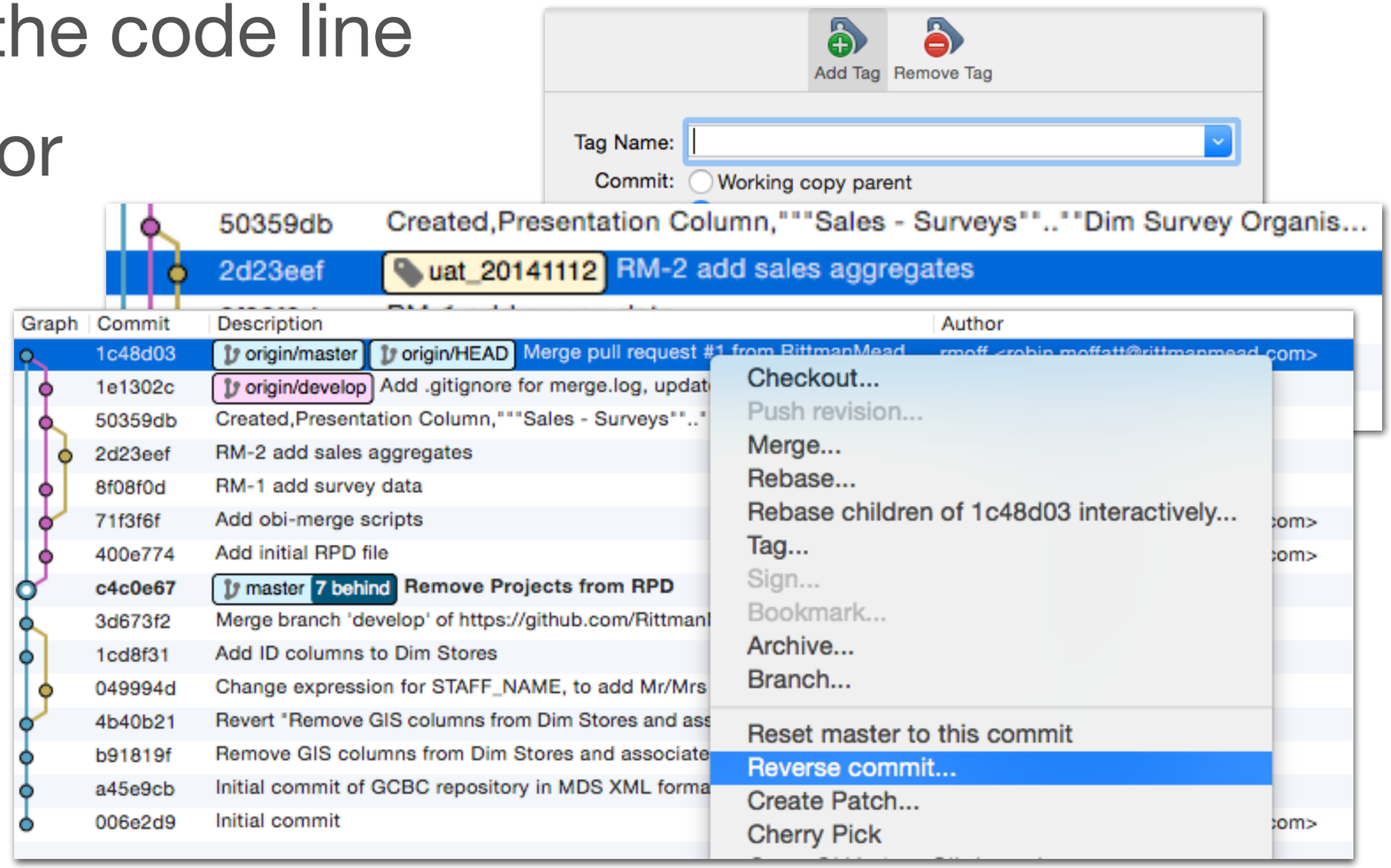
  - Pre-requisite to concurrent development method (discussed later)

Why Do We Need Source Control?



Why Do We Need Source Control?

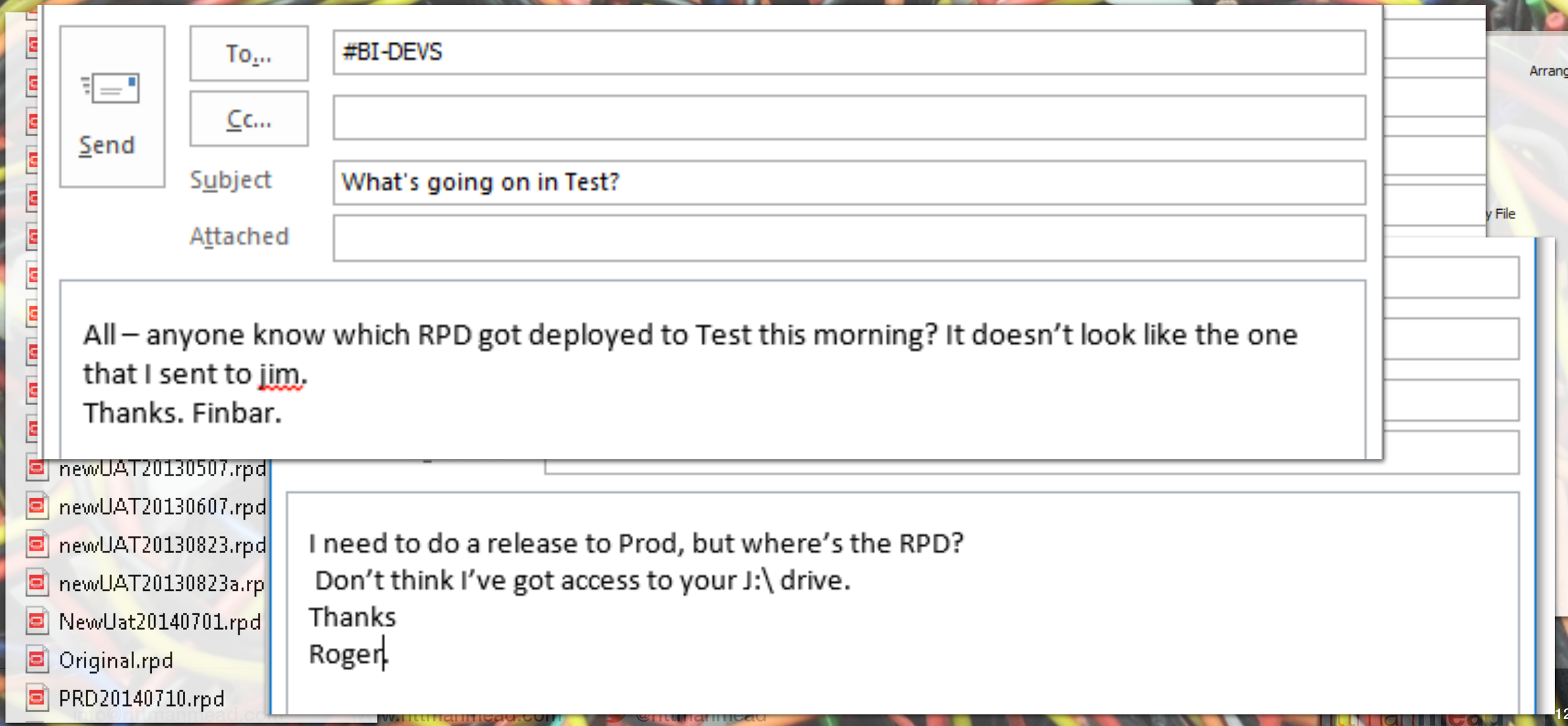
- Label (tag) points in the code line
- Mark code bundles for release to a given environment
  - Audit what went where when
- Revert to “known-good” point



Isn’t Source Control Scary and Complicated?

- No!
- Well… maybe a bit
- But you don’t really have a choice

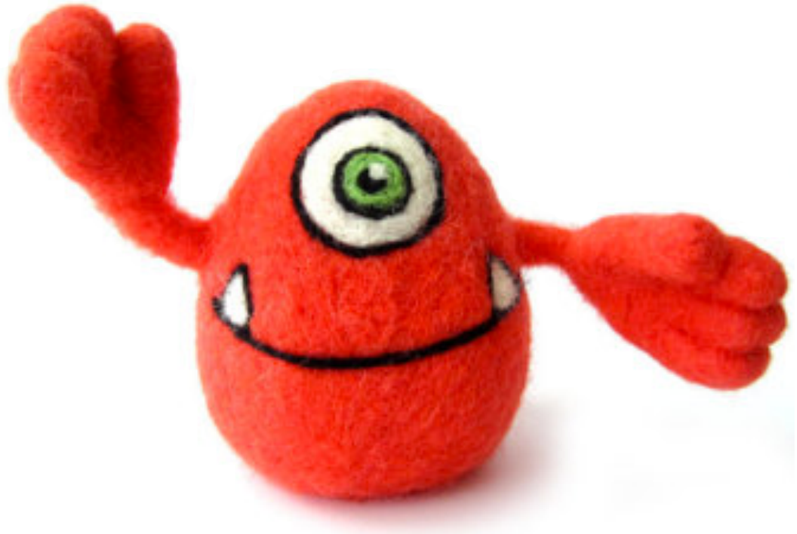
Without Source Control We Have Chaos!



Source Control is Mandatory

- Any mature OBIEE development should be using Source Control

- One developer or one hundred, still relevant



What Are the Tools for Source Control in OBIEE?

- Common ones (but plenty of others):
  - git
  - Subversion (SVN)
  - Perforce

- **Familiarity with the tool and organisational acceptance is key**

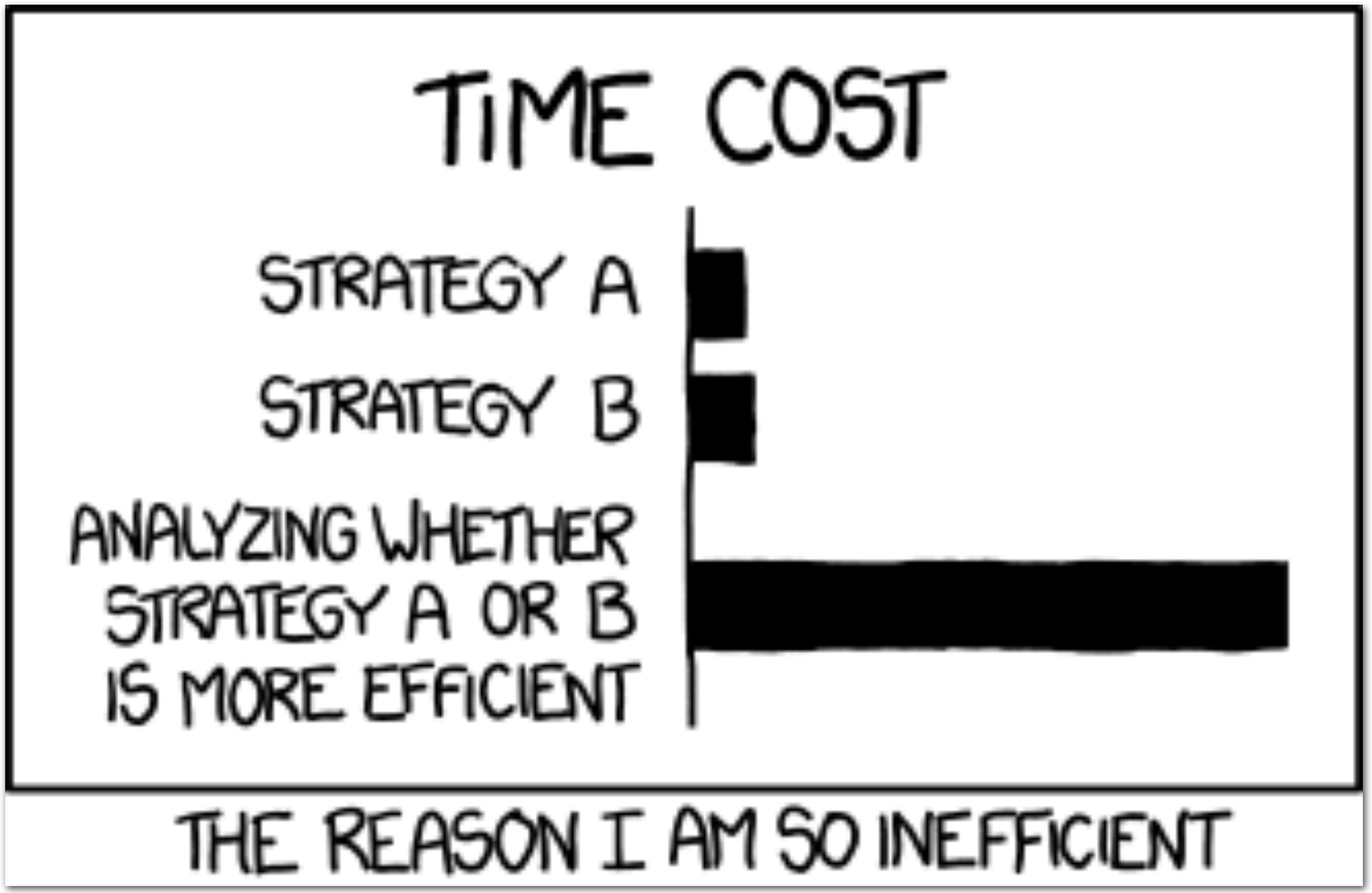
Our tool of choice? git

- Particularly if you are wanting to do concurrent development, **git** is excellent
  - Modern technology, and Distributed

- Self-host with **gitlab**, or SaaS with **github** or **bitbucket**

- **BUT** - Don’t rip out Subversion if already heavy invested

Just Use Source Control Already



OBIEE Artefacts in Source Control

- There are a variety of artefacts that we need to keep under source control
- What about the .bar file?
- How does it integrate with deployment practices?
- Time to dig deeper …

OBIEE 12c Code Control, Migration, and Deployment

OBIEE 12c Code Control and Deployment

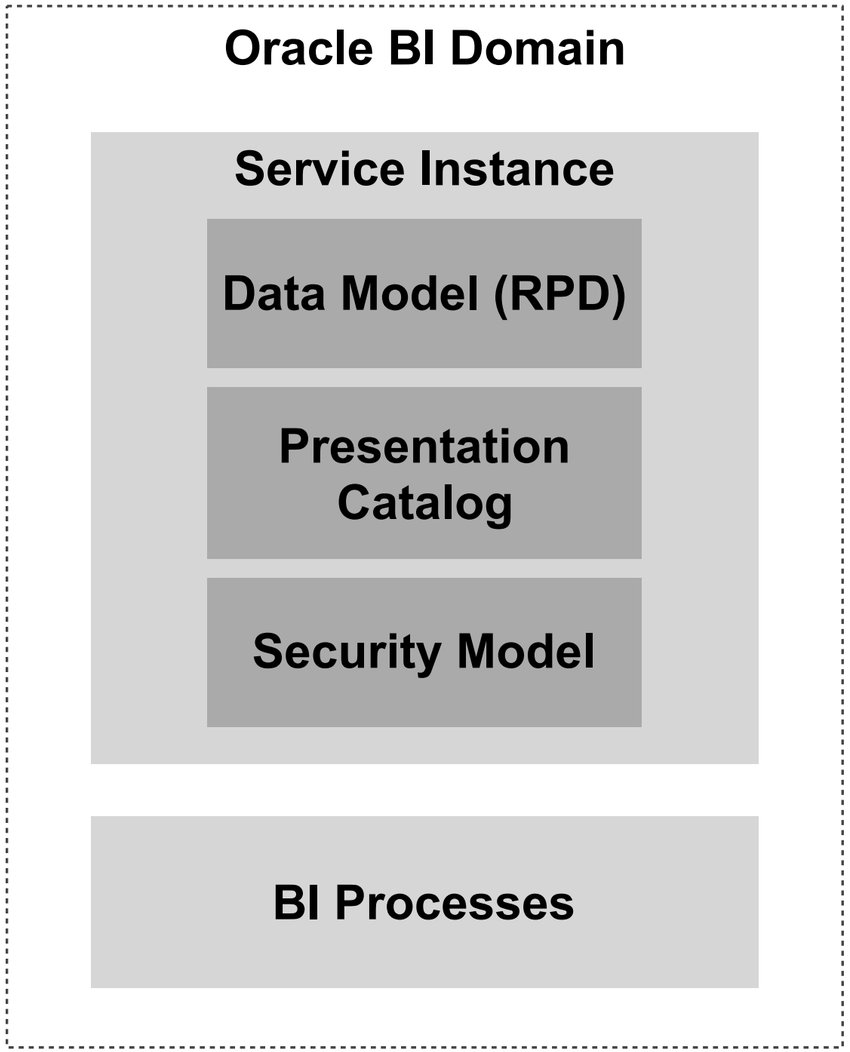
- Environment migration process is sometimes referred to as Test to Production (“T2P”)
- Source Control should always be the point from which code is deployed
- Deployment should always be automated
  - Faster
  - Reduces risk of human error
  - Easier to support

OBIEE 12c Code Control and Deployment

- Challenges:
  - RPD - embeded credentials
  - Presentation Catalog - often can’t lift & shift
    - User-developed content in Production, User saved preferences

OBIEE Service Instance

- OBIEE 12c has the concept of a “Service Instance”, containing the metadata needed for an ‘analytical application’:
  - Data Model (RPD)
  - Report and Dashboard *definitions* (Presentation Catalog)
  - Security (Application Role and Policy mappings and memberships)



OBIEE Code Artefacts

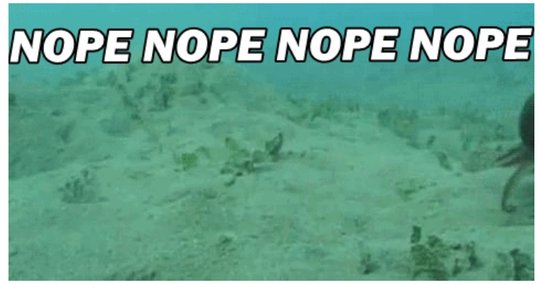
- Service Instances can be exported to a BI Archive (BAR) file using **exportServiceInstance**
- By default it *excludes*:
  - Connection pool credentials
  - Presentation Catalog /users and *some of* /system
- Available overrides:
  - `includeCredentials`
  - `includeCatalogRuntimeInfo`

The BAR file is not the whole answer!

- And here's why …

The RPD is Easy…

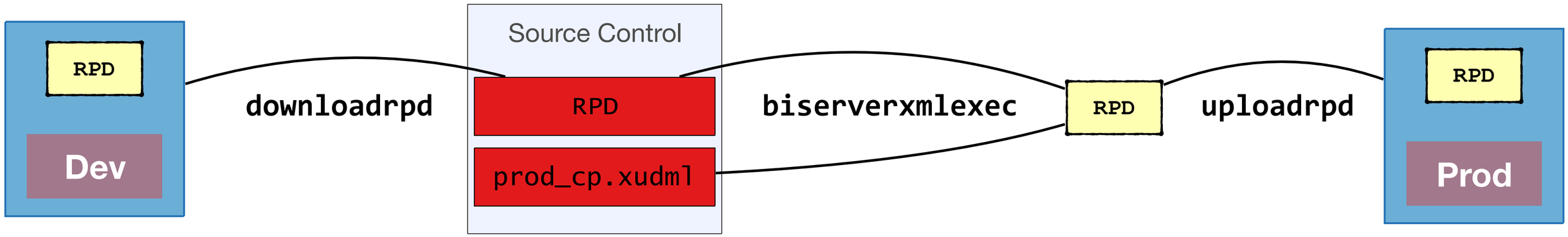
- Two deployment options
  - BAR
  - RPD only
- Store it as a Binary object
  - unless you want to dig into merging MDS XML … nope.



- All that should differ in target is Connection Pool credentials

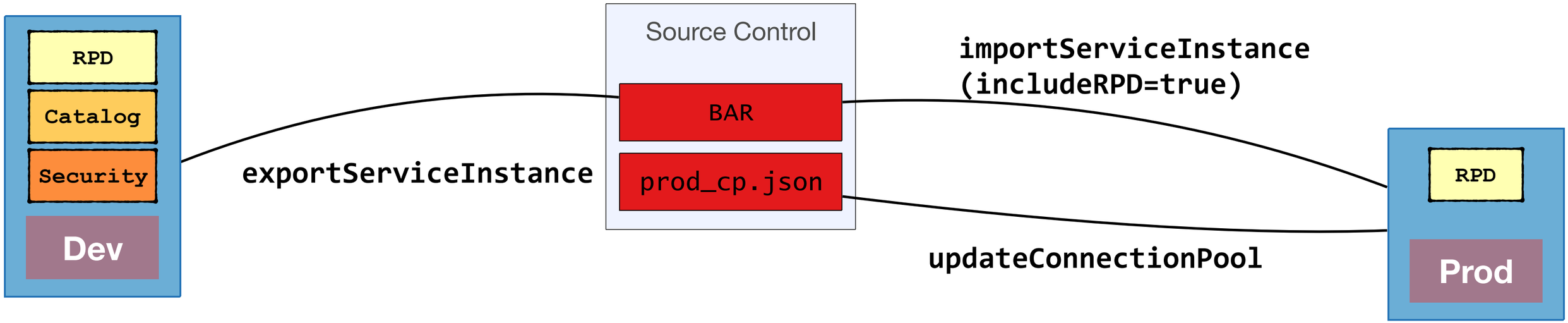
RPD option 1 - download/upload directly

- Create XUDML patch for target Connection Pool
  - `comparerpd`
  - Store RPD from development in Source Control
- Patch prior to deploy with updated Connection Pool
  - `biserverxmlexec`
- One step deploy: uploadrpd

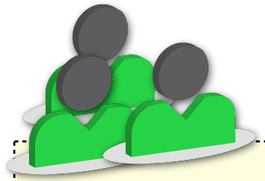


RPD option 2 - BAR

- Create JSON config for target Connection Pool
- Store BAR from development in Source Control
- Two-step deploy:
  - 1. Deploy BAR to target, restart BI Server.
  - 2. Once deployed, update connection pools with JSON



…the Presentation Catalog is Not Easy



Developer-created ‘pre-canned’ reports and dashboards

/shared/Finance/UK
/shared/HR
/shared/Sales

Test

/shared/Finance/UK
/shared/HR
/shared/Sales

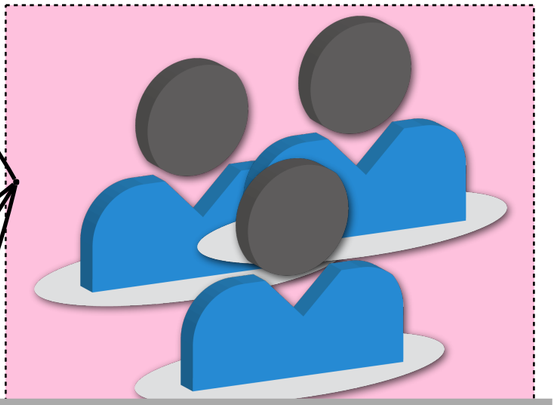
UAT

Governed release process

/shared/Finance/UK
/shared/Finance/Corp
/shared/HR
/shared/Sales
/users/rmellie/rep01
/users/rmellie/_internals
/system

Production

Ad-hoc creation in Prod

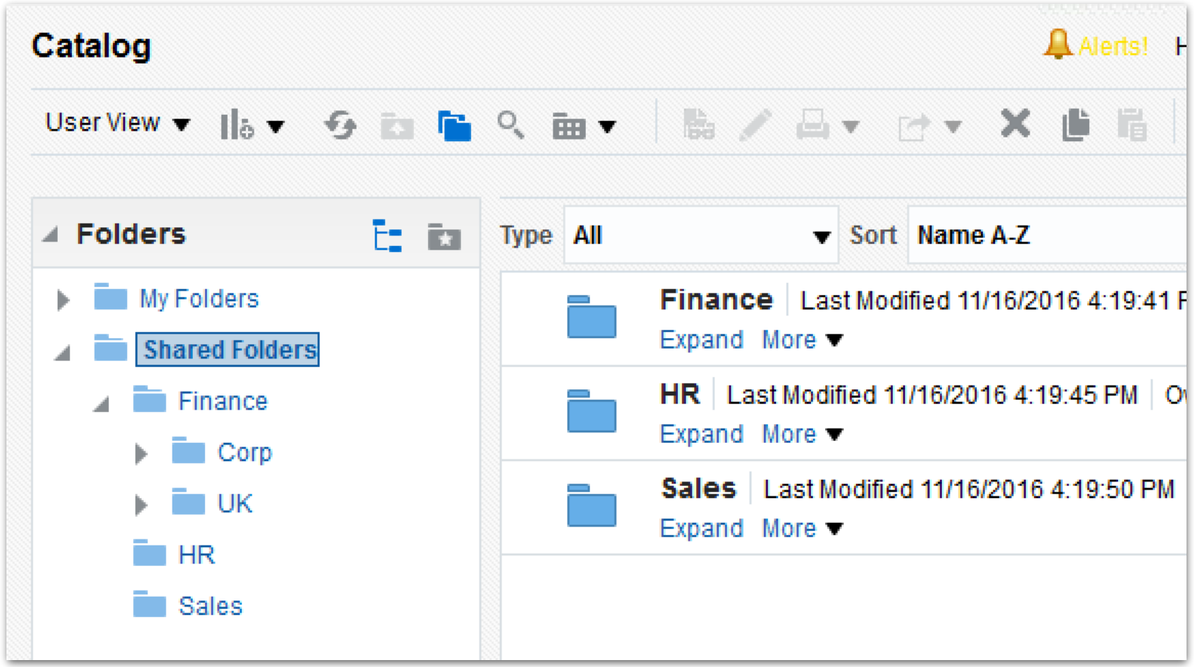


User runtime preferences

System settings

…the Presentation Catalog is Not Easy

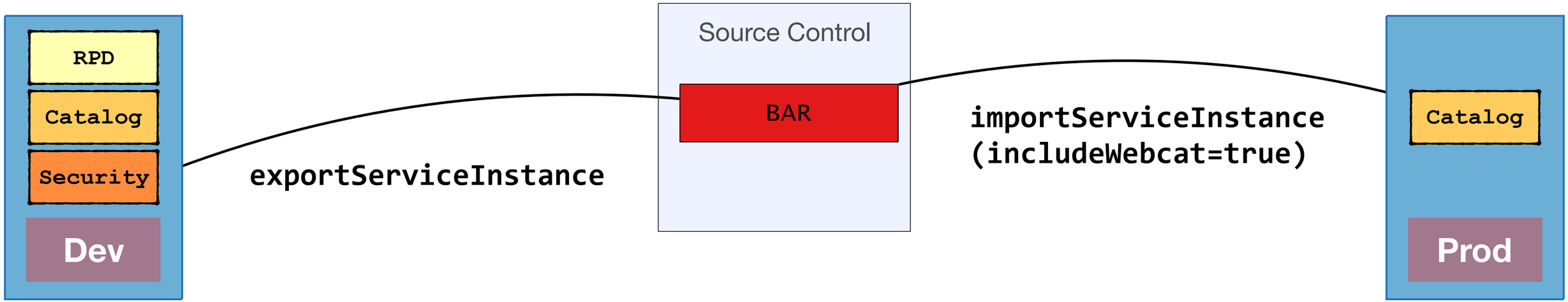
- Option 1 : BAR + .catalog /user export/import
- Option 2 : Baseline + .catalog deltas




```
├── system
|   ├── bookmarks
|   ├── briefingbooksnapshots
|   ├── cspwhitelist
|   ├── devices
|   ├── disconnected
|   ├── globalvariables
|   ├── logconfig
|   ├── metadata
|   ├── mktgcache
|   ├── mktgdefaults
|   ├── mktgjob
|   ├── privs
|   |   ├── actionprivs
|   |   ├── adminsystemprivs
|   |   ├── alertssystemprivs
|   |   ├── briefingbookprivs
|   |   ├── catalog
|   |   ├── catalogsystemprivs
|   |   ├── co%6editionprivs
|   |   ├── dashboardsystemprivs
|   |   ├── formatsystemprivs
|   |   ├── generalprivs
|   |   ├── home
|   |   ├── marketingsystemexportformatsprivs
|   |   ├── marketingsystemsegmentationprivs
|   |   ├── myaccountprivs
|   |   ├── proxyprivs
```

Presentation Catalog - BAR Export/Import

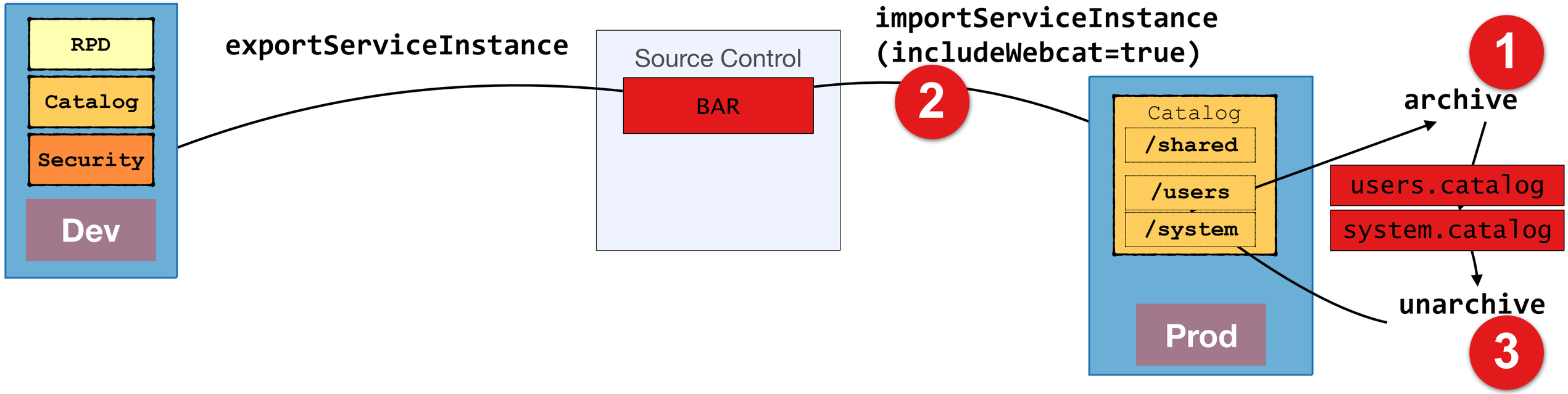
- **Replaces** the entire target catalog



- Overwrites **all user content** (My Folders), **user preferences** (dashboard customisations, favourites, etc)
- Overwrites **functionality privilege grants** to that of *source* environment

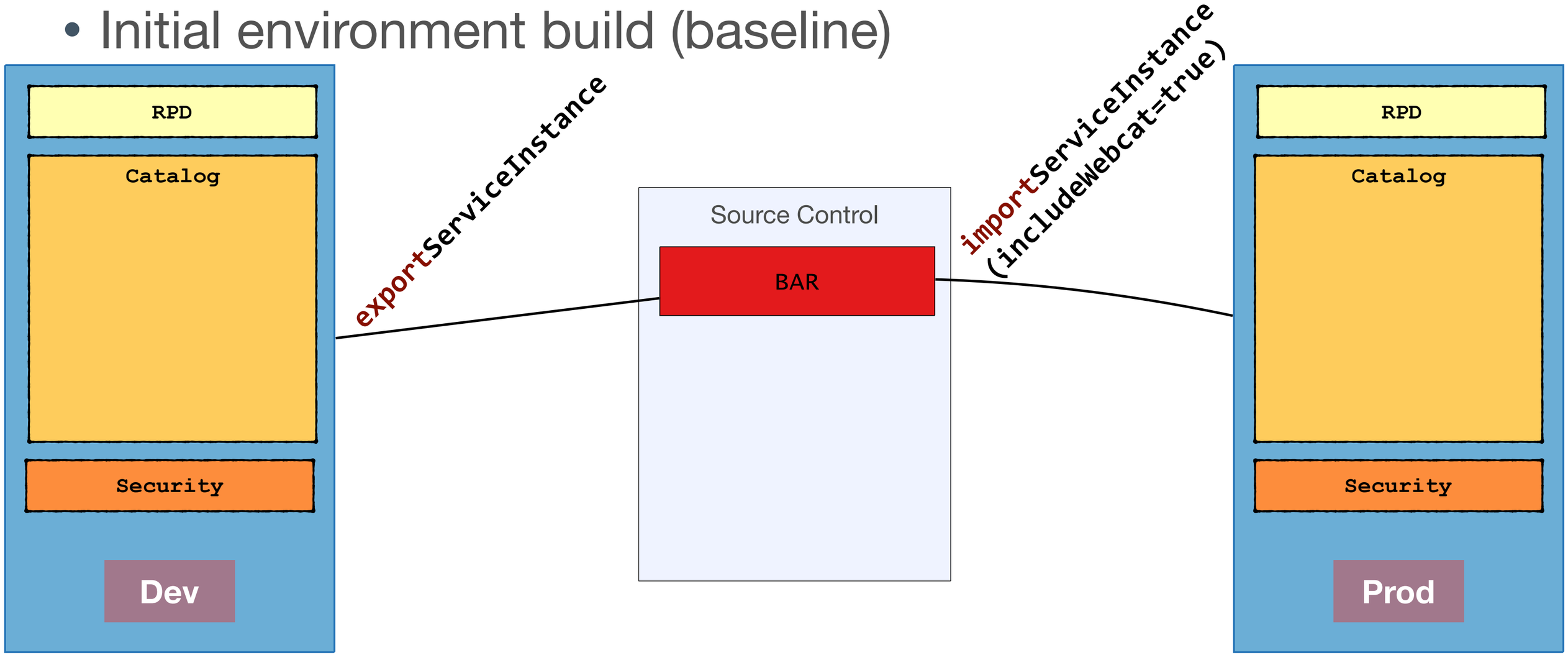
Presentation Catalog - Option 1 - BAR Export/Import

- Before deploying BAR, create .catalog export of /users and /system
- After deploying BAR, re-import /users and /system



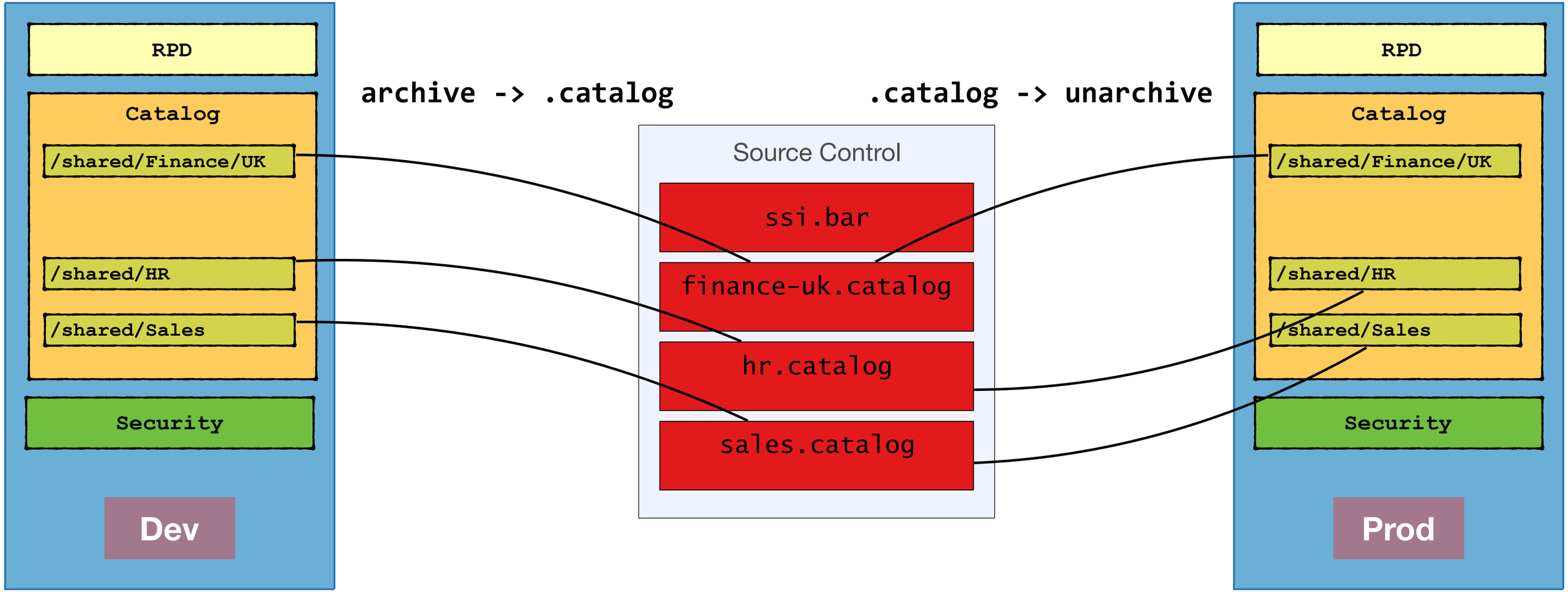
Presentation Catalog - Option 2 - BAR baseline + .catalog deltas

- Initial environment build (baseline)



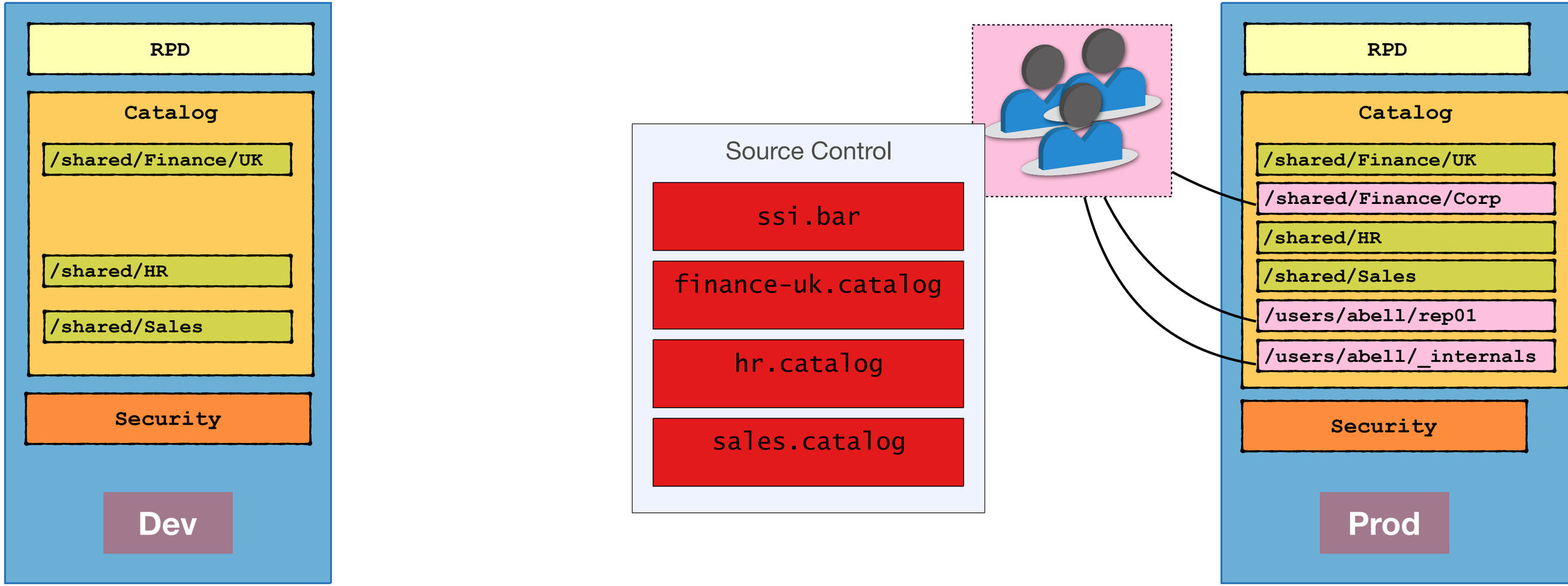
Presentation Catalog - Option 2 - BAR baseline + .catalog deltas

- .catalog archive/unarchive for subsequent development



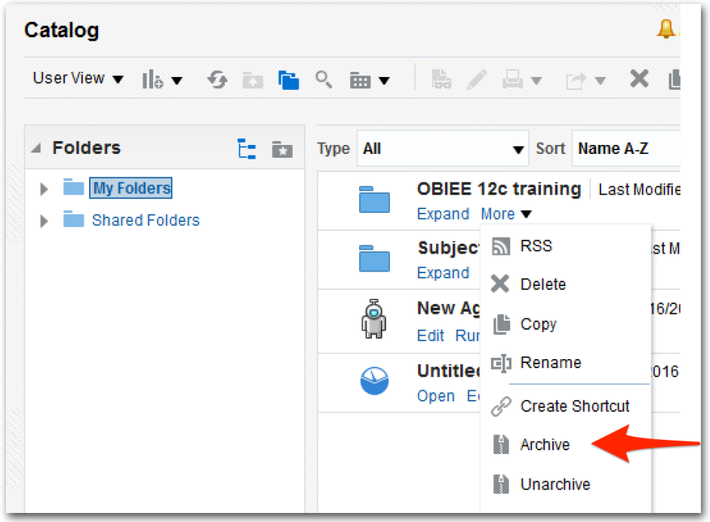
Presentation Catalog - Option 2 - BAR baseline + .catalog deltas

- User preferences and user-created content direct in Prod



Presentation Catalog Archive Files

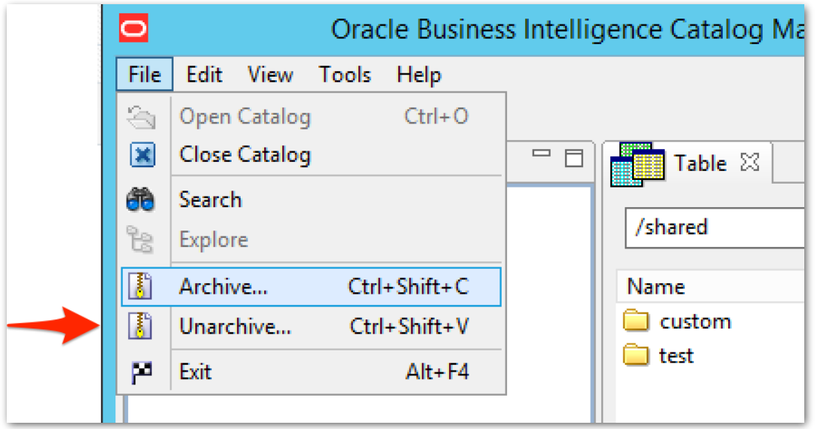
- Create the .catalog files
  - Manually
    - OBIEE front end
    - Catalog Manager GUI
  - Automatically
    - Catalog Manager CLI
    - WebCatalogService.copyItem2




```
runcat.sh \
-cmd archive  \
-online http://demo.us.oracle.com:7780/analytics/saw.dll \
-credentials /tmp/creds.txt \
-folder "/shared/HR" \
-outputFile /home/oracle/obiee12c-sdlc-demo/hr.catalog
```

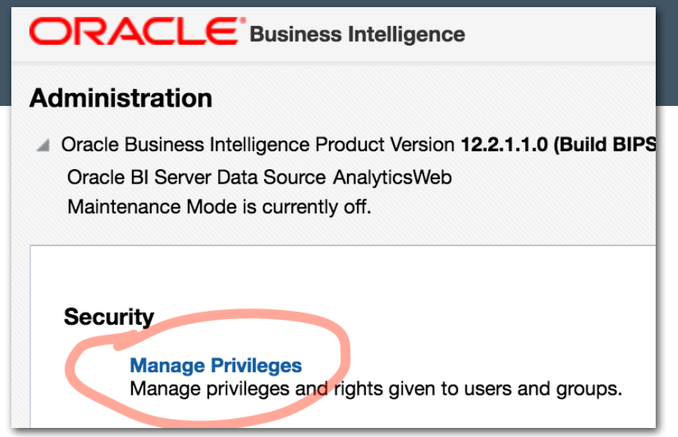
Presentation Catalog Archive Files

- Deploy to target environment
  - Manually
    - OBIEE front end
    - Catalog Manager GUI
  - Automatically
    - Catalog Manager CLI
    - WebCatalogService.pasteItem2




```
runcat.sh \
-cmd unarchive \
-inputFile hr.catalog \
-folder /shared \
-online http://demo.us.oracle.com:7780/analytics/saw.dll  \
-credentials /tmp/creds.txt \
-overwrite all
```

Security is Relatively Easy…



- Define security policy in Development
  - Data model security (RPD)
  - Application Roles and memberships, Application Policies
  - OBIEE permissions (access to catalog folder, privilege to use funcitonalities)
- Export to BAR (this includes all necessary security artefacts)
  - RPD
  - Catalog (/system/privs)
  - Policy Store
- Import to target environment
- Permission grants have to be *subsequently* managed in **target** environment

So How Do You Choose?

- Most flexible and can be used with concurrent development
  - RPD: .rpd + xudml target environment patch file
  - Catalog: BAR baseline + .catalog per folder from development
- Least flexible, most straightforward, no scope for concurrent development
  - RPD: BAR + json target environment connection pool config
  - Catalog: BAR + option target environment /user export&import

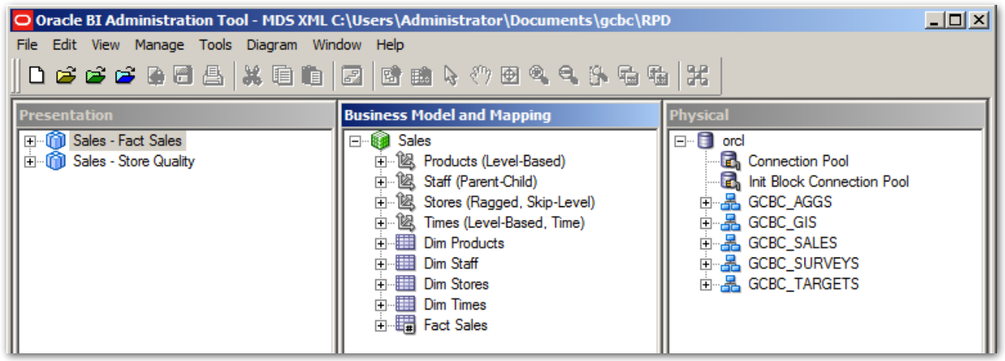
Source Control and Code Migration - Summary

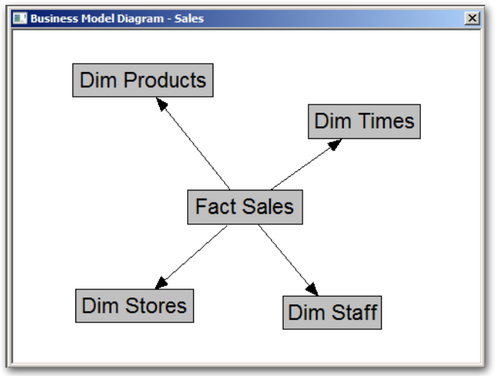
- Use Source Control
- Ideally, automate all RPD/Catalog deployments
- BAR is helpful, but not the whole story. Don’t forget
  - Component configuration
  - Custom skins & styles, writeback templates, etc.
  - Target environment connection pool settings (JSON/XUDML)

Concurrent RPD Development in OBIEE

The RPD

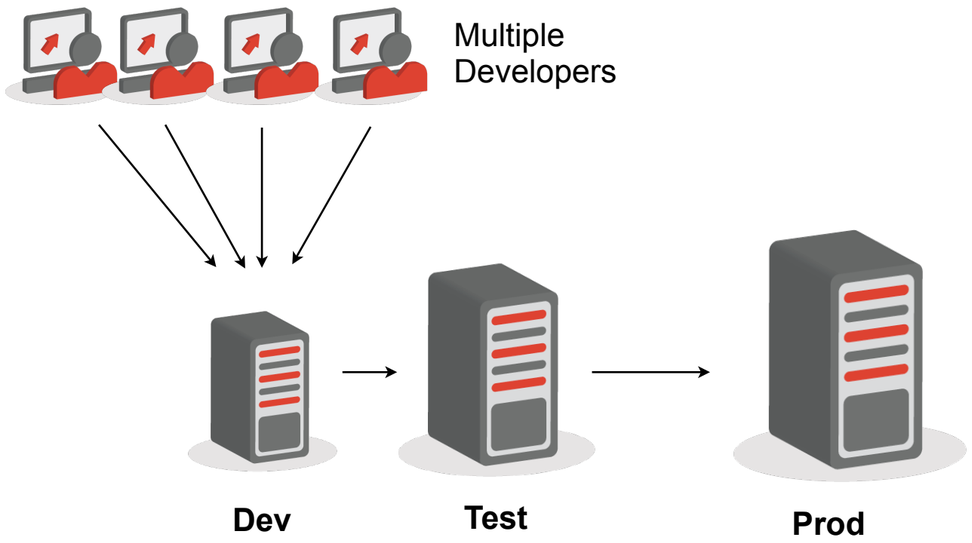
- At the core of OBIEE is the RPD - the brains of the operation!
- A metadata model that generates SQL requests at runtime from user report requests
- **Most concurrent OBIEE development problems centre around the RPD**





The Problem With Serial RPD Development

- It doesn't scale well
- Releases are complicated and risky



Feature Driven Development

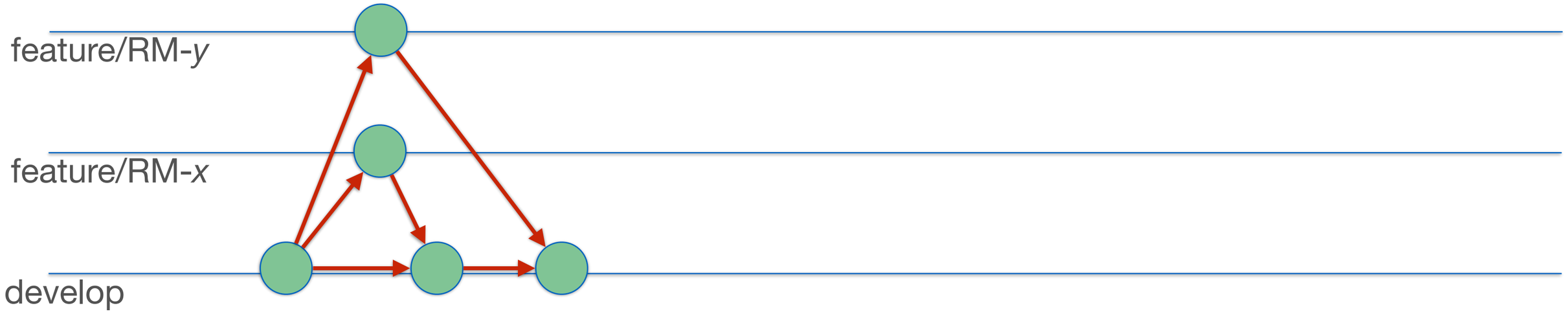
- Development is broken up into “Features”, such as:
  - New logical column
  - Add a dimension
  - Fix a bug
- Great for flexible (Agile/agile) approach to delivery
  - Features only released when ready
  - Features don't block others
  - Selectively release features

We've Got To Have Source Control!

- Any project should always be using **source control**
  - To do properly flexible concurrent development, we need to take a **feature-driven** approach

➡Thus we use **branch based source control**

➡But, how do we merge RPD changes between branches?



The Harsh Reality

- OBIEE does not natively support concurrent development integrated with source control
  - *And it never will until Oracle re-engineer OBIEE and/or the RPD storage format*

- All options require a **defined process to be followed** and associated **training** for staff
  - There is no perfect software-based solution that will **force 'bad' developers to be good**

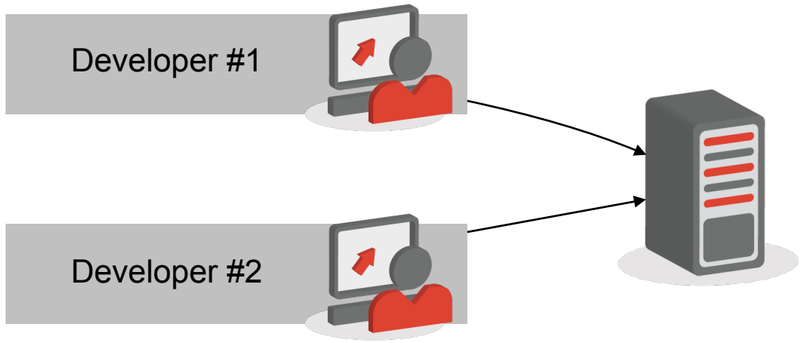
- You will always need a "**Dungeon Master**" to oversee and track developments, manage conflicts, and support developers.

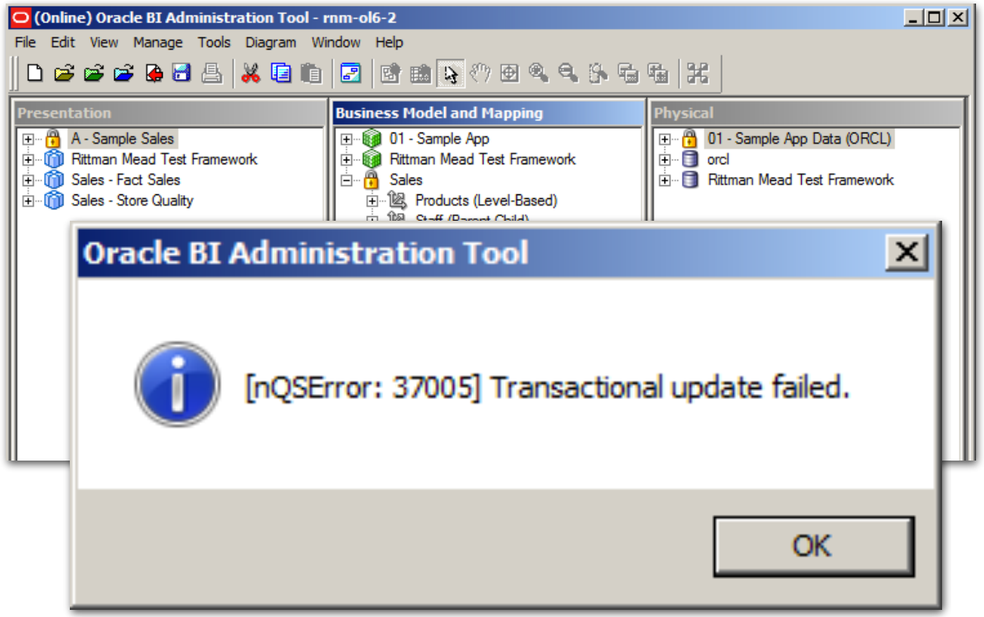
The Harsh Reality

- There are three (and a half) options to consider for concurrent development in OBIEE
  - Online editing
  - MUD
  - Source control + RPD merging
    - MDS XML or binary RPD?
- **They all suck!**
- **Which sucks the least?…**
  - What can we do to mitigate the suckiness?

Online RPD edits

- Easy to use - Zero setup required
- BUT
  - High-level locks taken when objects checked out
  - Can’t reverse changes except manually
  - Notoriously fragile - "Transactional update failed”
  - No source control integration

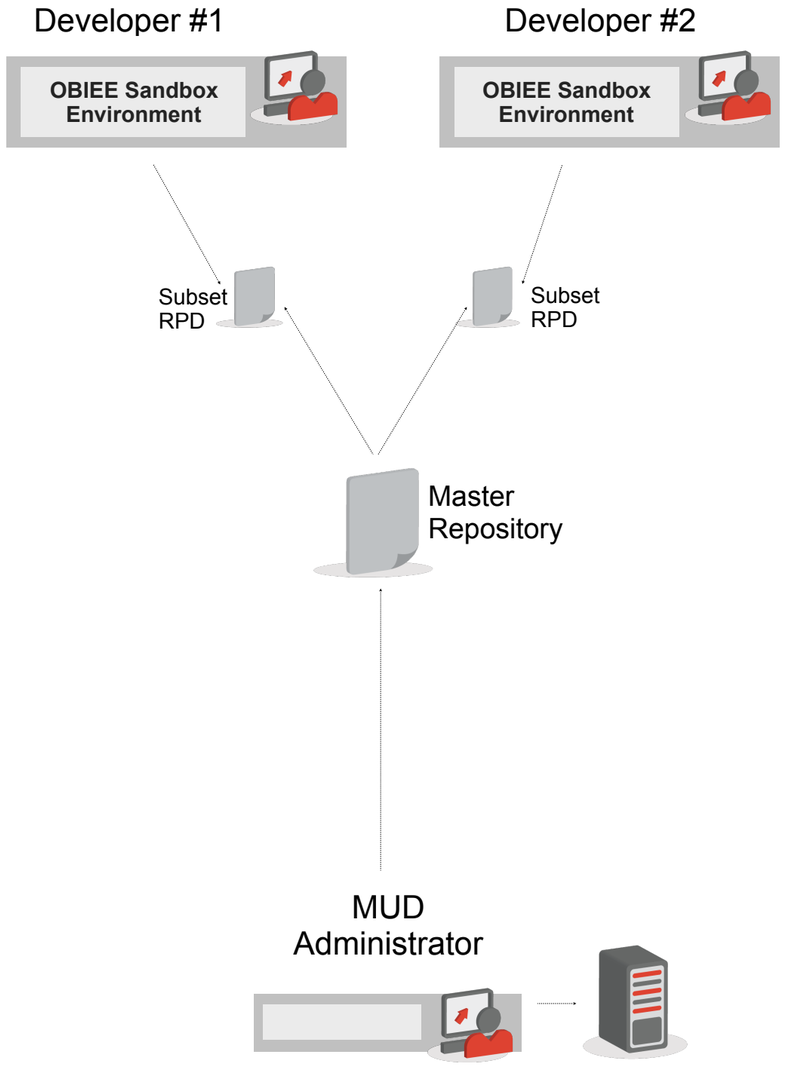




- Great for **individual** developers in isolation making sandbox changes
- Useless for **multiple** developers

Multi User Development Environment (MUD)

- Developers work on a subset of the master RPD, via a network share
- Completed changes merged back into master RPD
- Closest thing there is to native concurrent development functionality in OBIEE
- BUT
  - Responsibility and power for conflict resolution is with the developer, not source master
  - Known bugs (eg Variables, parent-child tables)
  - No native integration with source control
    - Can't backout changes
    - Doesn't easily support feature-driven development

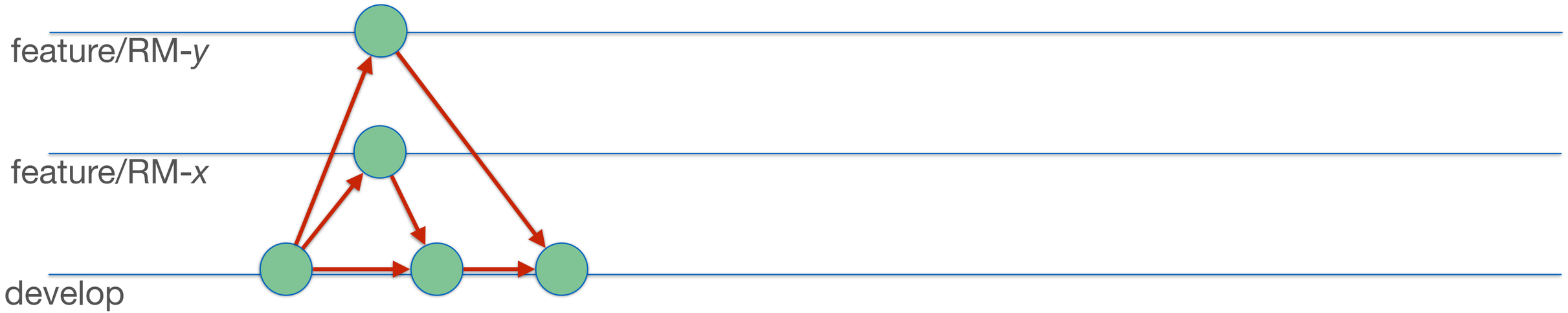


We've Got To Have Source Control!

- Any project should always be using **source control**
  - To do properly flexible concurrent development, we need to take a **feature-driven** approach

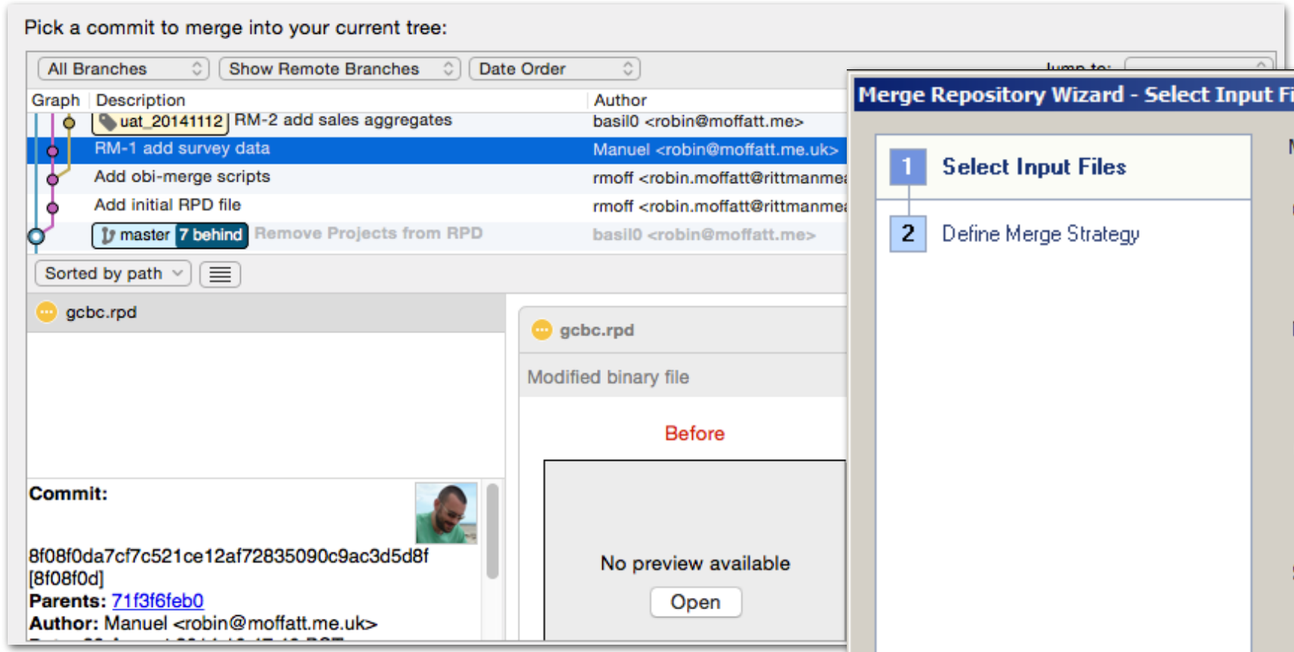
➡Thus we use **branch based source control**

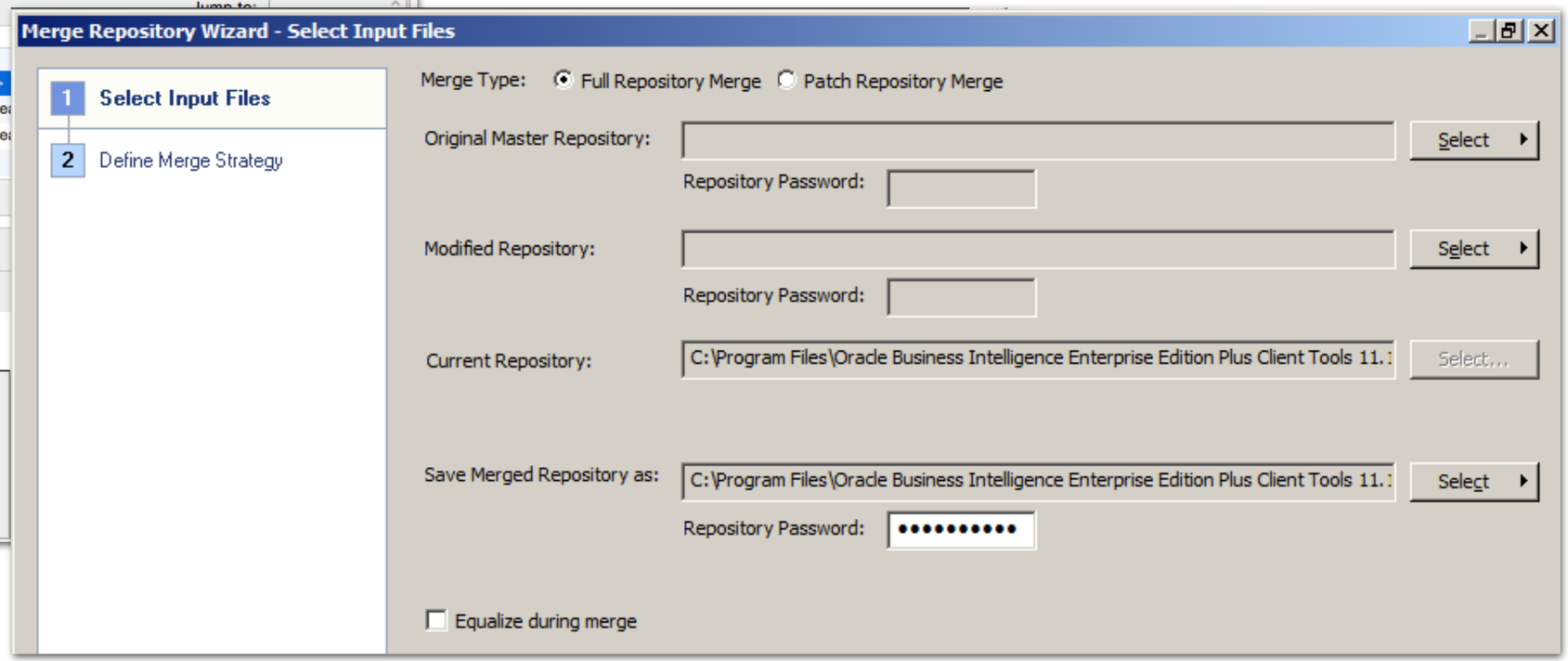
➡But, how do we merge RPD changes between branches?



Source Control Merging

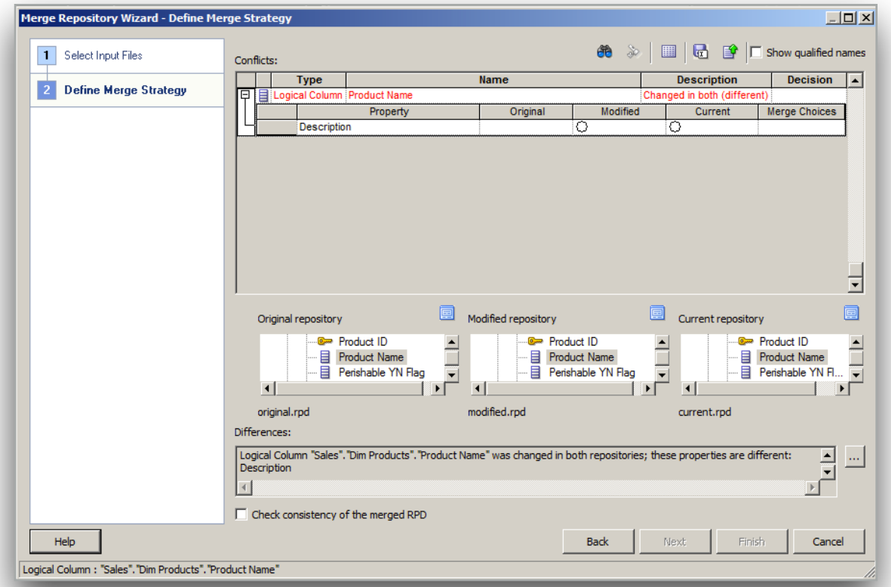
- When branches are combined in source control, the tool will automagically merge files and folders
- **Source control tools can only automatically merge files that are not binary…**



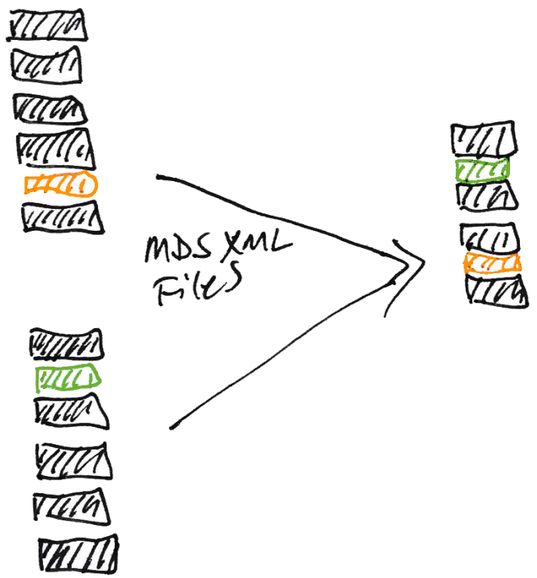


How Do We Merge The RPD?

- Whilst the RPD is logically comprised of multiple related objects, physically it is a **single binary file**
- Merging the binary RPD can only be done through the OBIEE Administration Tool's merge feature — **NOT source control tool**!
- Therefore we have to **merge the RPD manually** before merging branches in source control
- Our workflow becomes disrupted and manual (== error prone)



Is MDS XML the answer?



- The RPD is broken up into a hierarchical set of XML files representing object types such as Logical Tables, Logical Table Sources, etc
- Because there are multiple plain text files, can the source control tool can carry out the merge itself?

```
<?xml version="1.0" encoding="UTF-8" ?>
<LogicalComplexJoin mdsid="m637d53ce-0cbd-1000-9eed-c0a8f8880000"
    name="Sales:Relationship_2004:9581293467713"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.oracle.com/obis/repository"
    type="Inner" logicalTable1Ref="/oracle/bi/server/base/LogicalTable/637d538e-0cbd-1000-9eed-c0a8f8880000.xml#m637d538e-0cbd-1000-9ee
    multiplicity1="0..1"
    isAggregate1="false"
    logicalTable2Ref="/oracle/bi/server/base/LogicalTable/637d5376-0cbd-1000-9eed-c0a8f8880000.xml#m637d5376-0cbd-1000-9eed-c0a8f888000
    multiplicity2="0..n" isAggregate2="false">
</LogicalComplexJoin>
```

```
drwxr-xr-x   3 robin  staff   102 29 Aug 10:51 BusinessModel
drwxr-xr-x   4 robin  staff   136 29 Aug 10:51 ConnectionPool
drwxr-xr-x   3 robin  staff   102 29 Aug 10:51 Database
drwxr-xr-x   7 robin  staff   238 29 Aug 10:51 Dimension
drwxr-xr-x   6 robin  staff   204 29 Aug 10:51 Group
drwxr-xr-x  45 robin  staff  1530 29 Aug 10:51 InitBlock
drwxr-xr-x   9 robin  staff   306  7 Jan 09:59 LogicalComplexJoin
drwxr-xr-x   9 robin  staff   306 29 Aug 10:51 LogicalTable
drwxr-xr-x  17 robin  staff   578 29 Aug 10:51 LogicalTableSource
drwxr-xr-x  39 robin  staff  1326 29 Aug 10:51 PhysicalTable
drwxr-xr-x   4 robin  staff   136 29 Aug 10:51 PresentationCatalog
drwxr-xr-x  11 robin  staff   374 29 Aug 10:51 PresentationHierarchy
drwxr-xr-x  11 robin  staff   374 29 Aug 10:51 PresentationTable
drwxr-xr-x   5 robin  staff   170 29 Aug 10:51 PrivilegePackage
drwxr-xr-x   3 robin  staff   102 29 Aug 10:51 QueryObject
drwxr-xr-x   7 robin  staff   238 29 Aug 10:51 Schema
drwxr-xr-x   3 robin  staff   102 29 Aug 10:51 User
drwxr-xr-x  57 robin  staff  1938 29 Aug 10:51 Variable
robin@RNMMBP ~/git/gcbc/RPD/oracle/bi/server/base $ cd LogicalTable
robin@RNMMBP ~/git/gcbc/RPD/oracle/bi/server/base/LogicalTable $ ll
total 112
-rw-r--r--  1 robin  staff  8108 29 Aug 10:51 637d5376-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  8408 29 Aug 10:51 637d537c-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  6617 29 Aug 10:51 637d5386-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  7070 29 Aug 10:51 637d538e-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  5965 29 Aug 10:51 637d5395-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  2555 29 Aug 10:51 637d539c-0cbd-1000-9eed-c0a8f8880000.xml
-rw-r--r--  1 robin  staff  6888 29 Aug 10:51 637d539f-0cbd-1000-9eed-c0a8f8880000.xml
robin@RNMMBP ~/git/gcbc/RPD/oracle/bi/server/base/LogicalTable $
```

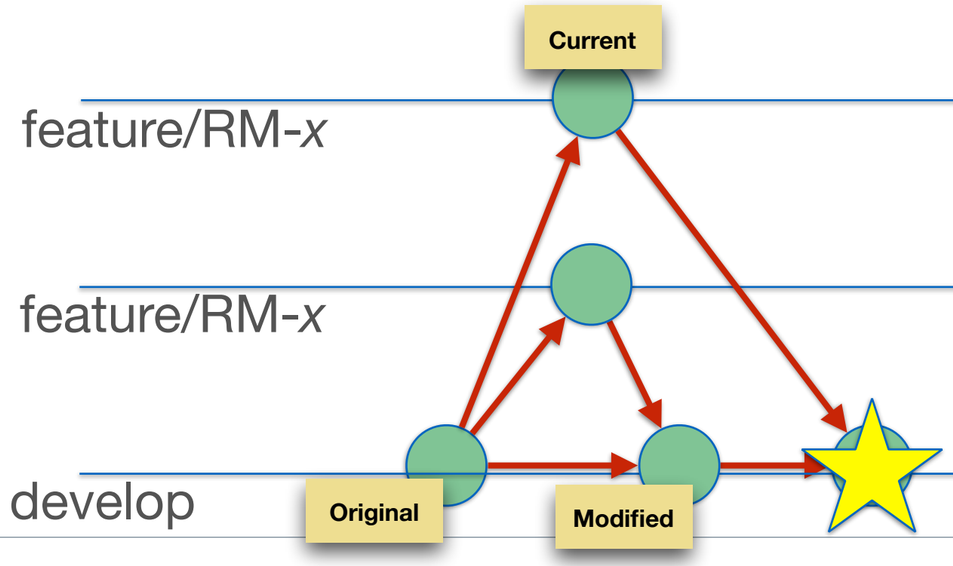
Is MDS XML the answer? No.

- **You cannot reliably merge the RPD except through Oracle's own tools**
  - MDS XML, whilst plaintext, is still **structured**
    - Unnecessary conflicts when merging & Potential inconsistencies / corruption
  - MDS XML uses GUIDs to name files, so source control may inadvertently **duplicate objects without notification**
- MDS XML merging may work for simple isolated changes to individual objects; as a development strategy this is restrictive.
- Read more:
  - https://www.rittmanmead.com/blog/2015/01/concurrent-rpd-development-in-obiee/
  - https://medium.com/red-pill-analytics/concurrent-rpd-development-with-obiee-the-checkmate-approach-8bd8279ec1d9#.ujccjg3fk

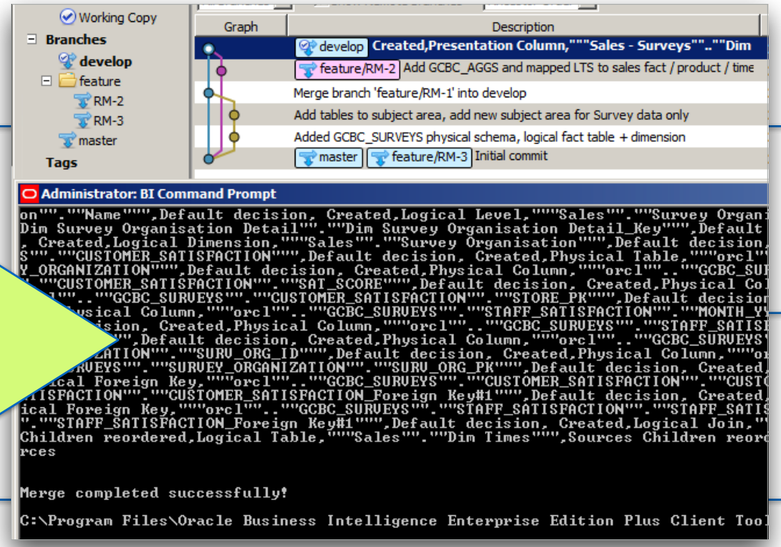
Source control + Binary RPD

1. Online edit
2. MUD
3. Source control + MDS XML RPD
4. **Source control + Binary RPD**

- Dedicated standalone developer sandbox environments
- Package up all artefacts in Source Control, providing release and rollback capabilities
- Use the common **git-flow** branching method, with development divided up into **features**
  - Provides clear structure and process for managing branches and releases
- RPD stored as binary. **Three-way merge with <u>OBIEE tools</u>** done where necessary.

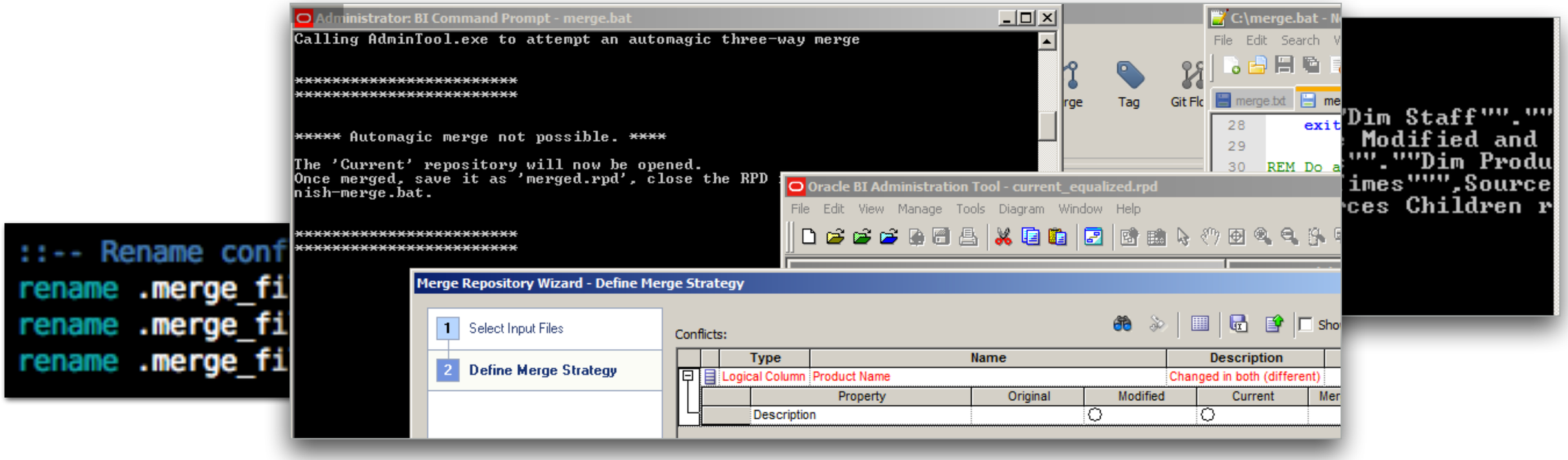


‘Secret Sauce’ :
merge.bat script



Source control + Binary RPD

- Conflict resolution script:
  - Uses git-flow to determine 3-way merge candidates
  - Attempts automatic merge (comparerpd/patchrpd)
  - Launches Administration Tool for user interaction if necessary

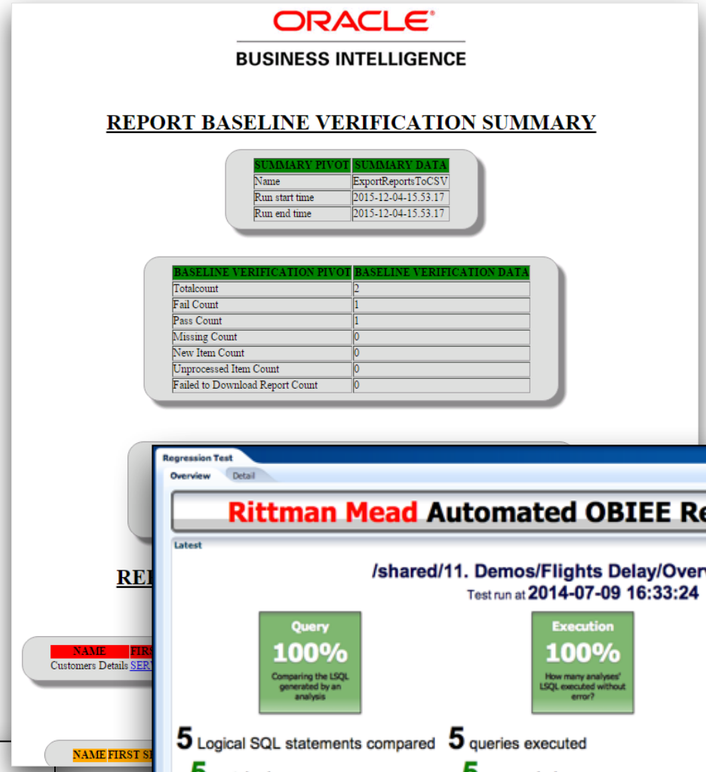


So which methods sucks the least?

	Multiple developers	Integration with source control	Feature driven development	Ease of use	Overall
Source control + Binary RPD	Yes	Yes	Yes	Medium	Not perfect, but the best option
MUD	Yes	Manual	No	Medium	Could be worse
Source control + MDS XML RPD	Yes	Yes	Only if objects fully isolated	Medium	Requires too much process to mitigate against RPD corruption and unnecessary merge conflicts
Online editing	Not really	No	No	Easy	Not viable

Checking It All Still Works

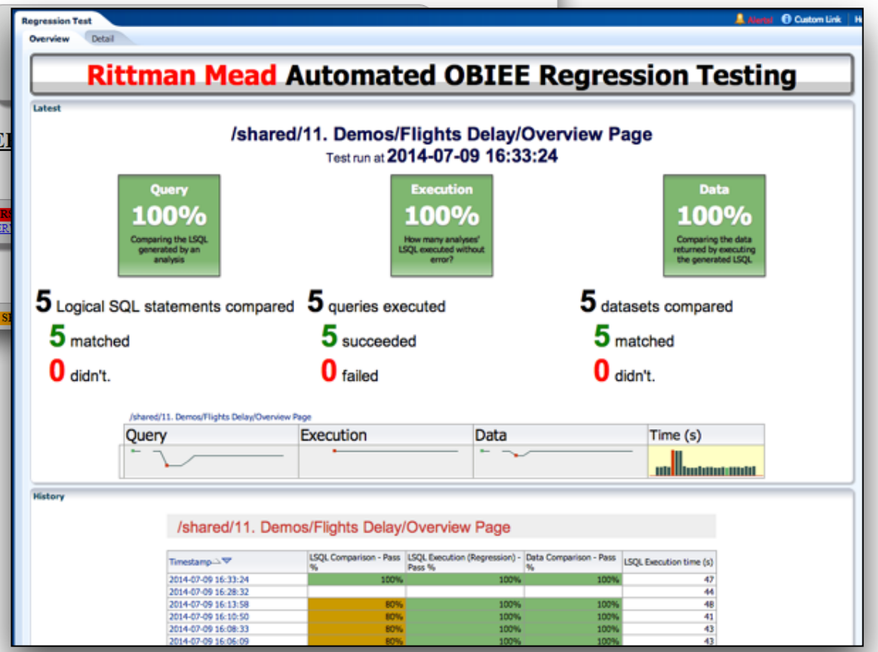
- For concurrent development to be successful, you need **automated testing**
  - **Baseline Validation Tool**
  - **Rittman Mead OBIEE Regression Testing Tool**


ORACLE
BUSINESS INTELLIGENCE

REPORT BASELINE VERIFICATION SUMMARY

SUMMARY PIVOT	SUMMARY DATA
Name	ExportReportsToCSV
Run start time	2015-12-04-15.53.17
Run end time	2015-12-04-15.53.17

BASELINE VERIFICATION PIVOT	BASELINE VERIFICATION DATA
Totalcount	2
Fail Count	1
Pass Count	1
Missing Count	0
New Item Count	0
Unprocessed Item Count	0
Failed to Download Report Count	0




```
Executing Logical SQL now...
Baseline data found. Capturing & comparing report data now.

[  OK  ]        Cost_Breakdown_Report--03_Pivot_Table_View
[  OK  ]        Overview--04_Chart_Views
[FAILED]        Overview--07_Narrative_View             nqcmd failed. Please examine regr/data/Overview--07_Narrative_View.out
[  OK  ]        Overview--07_Ticker_View
[  OK  ]        Performance_Breakdown--05_Table_with_conditional_formatting
[  OK  ]        Performance_Breakdown--06_Column_Selector
[  OK  ]        Performance_Breakdown--06_View_Selector


      Passed: 6       Warnings: 0     Failed: 1
```

Summary

- **If you're not doing source control, go and do it.**
  - Now.
  - Even for serial development
  - Even if you're the only developer
- Plan, and automate, your **code deployment** strategy, as this will drive your source control implementation
- The only way to do **safe and reliable concurrent development** is using the **binary RPD** stored in source control such as **git**

#EOF

speakerdeck.com/rmoff/

email

robin@rittmanmead.com

web

http://ritt.md/rmoff

http://rmoff.net

twitter

@rmoff

irc

rmoff @ #obihackers